Day 7 Task: Understanding package manager and systemctl

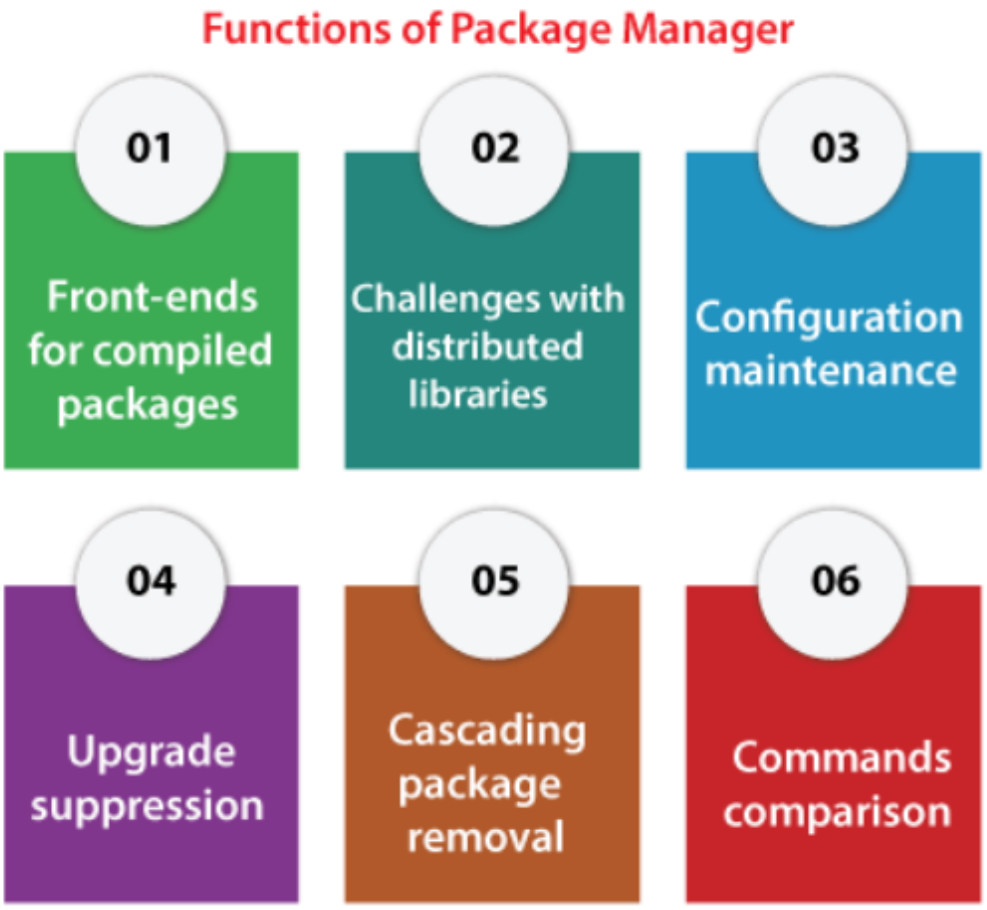
What is a package manager in Linux?

In simpler words, a package manager is a tool that allows users to install, remove, upgrade, configure and manage software packages on an operating system.

Packages include metadata like the name of the software, description of its objective, checksum (a cryptographic hash function preferably), dependency list, vendor, and version number essential for the software to properly run.

The package manager can be a graphical application like a software centre or a command line tool like apt-get or pacman.

You'll often find me using the term 'package' in tutorials and articles, To understand package manager, you must understand what a package is.



Pic credit: Javatpoint

What is a package?

A package is usually referred to an application but it could be a GUI application, command line tool or a software library (required by other software programs). A package is essentially an archive file containing the binary executable, configuration file and sometimes information about the dependencies.

Different kinds of package managers

Package Managers differ based on packaging system but same packaging system may have more than one package manager.

For example, RPM has Yum and DNF package managers. For DEB, you have apt-get, aptitude command line based package managers.

Tasks

1. You have to install docker and jenkins in your system from your terminal using package managers
2. Write a small blog or article to install these tools using package managers on Ubuntu and CentOS

Installing Docker:

Docker is a platform for running applications in an isolated environment called a "container" (or Docker container). Applications like Jenkins can be downloaded as read-only "images" (or Docker images), each of which is run in Docker as a container. A Docker container is a "running instance" of a Docker image. A Docker image is stored permanently, based on when image updates are published, whereas containers are stored temporarily.

To install docker, I will use the AWS Cloudshell. With the help of official docker documentation, we can easily follow the installation procedure here:

`sudo yum update -y`command in Linux is used on Red Hat-based distributions such as CentOS and Fedora to update the installed packages to the latest available versions. Here's what each part of the command does:

- `sudo`: Executes the command with superuser (root) privileges, allowing you to make system-wide changes.

- `yum`: The package management tool used in Red Hat-based Linux distributions for installing, updating, and removing packages.
- `update`: This sub-command instructs `yum` to update all installed packages to their latest versions.
- `-y`: Automatically confirms any prompts that ask for confirmation to install or update packages. It is used to skip the need for manual confirmation during the update process.

```
[ec2-user@ip-172-31-28-113 ~]$ sudo yum update -y
Last metadata expiration check: 0:08:38 ago on Wed Nov  8 10:22:42 2023.
Dependencies resolved.
Nothing to do.
Complete!
```

This command uses `apt-get` to install the following packages:

- `ca-certificates`: This package contains the public certificates of Certificate Authorities (CAs). It is essential for secure communication over HTTPS.
- `curl`: A command-line tool for transferring data with URLs. It supports various protocols, including HTTP, HTTPS, FTP, FTPS, SCP, SFTP, LDAP, and more.
- `gnupg`: The GNU Privacy Guard, a tool for secure communication and data encryption. It is commonly used for encrypting and signing emails, files, and other data.

sudo yum -y install docker

```
[ec2-user@ip-172-31-28-113 ~]$ sudo yum -y install docker
Last metadata expiration check: 0:12:54 ago on Wed Nov  8 10:22:42 2023.
Dependencies resolved.
================================================================================
 Package                 Arch     Version                   Repository     Size
================================================================================
Installing:
 docker                  x86_64   24.0.5-1.amzn2023.0.2     amazonlinux    42 M
Installing dependencies:
 containerd              x86_64   1.7.2-1.amzn2023.0.4      amazonlinux    34 M
 iptables-libs           x86_64   1.8.8-3.amzn2023.0.2      amazonlinux   401 k
 iptables-nft            x86_64   1.8.8-3.amzn2023.0.2      amazonlinux   183 k
 libcgroup               x86_64   3.0-1.amzn2023.0.1        amazonlinux    75 k
```

```
Installed:
  containerd-1.7.2-1.amzn2023.0.4.x86_64
  docker-24.0.5-1.amzn2023.0.2.x86_64
  iptables-libs-1.8.8-3.amzn2023.0.2.x86_64
  iptables-nft-1.8.8-3.amzn2023.0.2.x86_64
  libcgroup-3.0-1.amzn2023.0.1.x86_64
  libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64
  libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64
  libnftnl-1.2.2-2.amzn2023.0.2.x86_64
  pigz-2.5-1.amzn2023.0.3.x86_64
  runc-1.1.7-1.amzn2023.0.3.x86_64

Complete!
```

sudo service docker start

Access Docker commands in `ec2-user` user

sudo usermod -a -G docker ec2-user

sudo chmod 666 /var/run/docker.sock

docker version

```
[ec2-user@ip-172-31-28-113 ~]$ sudo systemctl start docker
[ec2-user@ip-172-31-28-113 ~]$ sudo systemctl enable docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /us
r/lib/systemd/system/docker.service.
```

Docker –version

```
[ec2-user@ip-172-31-28-113 ~]$ docker --version
Docker version 24.0.5, build ced0996
```

```
[ec2-user@ip-172-31-28-113 ~]$ sudo usermod -a -G docker ec2-user
[ec2-user@ip-172-31-28-113 ~]$ sudo chmod 666 /var/run/docker.sock
[ec2-user@ip-172-31-28-113 ~]$ docker version
Client:
 Version:           24.0.5
 API version:       1.43
 Go version:        go1.20.10
 Git commit:        ced0996
 Built:             Fri Oct 13 00:00:00 2023
 OS/Arch:           linux/amd64
 Context:           default

Server:
 Engine:
```

systemctl and systemd

systemctl is used to examine and control the state of “systemd” system and service manager. systemd is system and service manager for Unix like operating systems(most of the distributions, not all).

Tasks

1. check the status of docker service in your system (make sure you completed above tasks, else docker won't be installed)
2. stop the service jenkins and post before and after screenshots
3. read about the commands systemctl vs service

eg. `systemctl status docker` vs `service docker status`

`systemctl status docker`

```
[ec2-user@ip-172-31-28-113 ~]$ systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: d>
     Active: active (running) since Wed 2023-11-08 10:39:54 UTC; 9min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 26814 (dockerd)
      Tasks: 9
     Memory: 25.9M
        CPU: 364ms
     CGroup: /system.slice/docker.service
             └─26814 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/con>

Nov 08 10:39:54 ip-172-31-28-113.us-west-2.compute.internal systemd[1]: Startin>
Nov 08 10:39:54 ip-172-31-28-113.us-west-2.compute.internal dockerd[26814]: tim>
Nov 08 10:39:54 ip-172-31-28-113.us-west-2.compute.internal dockerd[26814]: tim>
Nov 08 10:39:54 ip-172-31-28-113.us-west-2.compute.internal dockerd[26814]: tim>
Nov 08 10:39:54 ip-172-31-28-113.us-west-2.compute.internal dockerd[26814]: tim>
Nov 08 10:39:54 ip-172-31-28-113.us-west-2.compute.internal dockerd[26814]: tim>
```

Service docker status

```
[ec2-user@ip-172-31-28-113 ~]$ service docker status
Redirecting to /bin/systemctl status docker.service
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: d>
     Active: active (running) since Wed 2023-11-08 10:39:54 UTC; 11min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 26814 (dockerd)
      Tasks: 9
     Memory: 25.9M
        CPU: 375ms
     CGroup: /system.slice/docker.service
             └─26814 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/con>

Nov 08 10:39:54 ip-172-31-28-113.us-west-2.compute.internal systemd[1]: Startin>
Nov 08 10:39:54 ip-172-31-28-113.us-west-2.compute.internal dockerd[26814]: tim>
Nov 08 10:39:54 ip-172-31-28-113.us-west-2.compute.internal dockerd[26814]: tim>
Nov 08 10:39:54 ip-172-31-28-113.us-west-2.compute.internal dockerd[26814]: tim>
Nov 08 10:39:54 ip-172-31-28-113.us-west-2.compute.internal dockerd[26814]: tim>
```

Jenkins

```
sudo wget -O /etc/yum.repos.d/jenkins.repo \
    https://pkg.jenkins.io/redhat-stable/jenkins.repo
```

```
[ec2-user@ip-172-31-28-113 ~]$ sudo wget -O /etc/yum.repos.d/jenkins.repo \
    https://pkg.jenkins.io/redhat-stable/jenkins.repo
--2023-11-08 10:56:29--  https://pkg.jenkins.io/redhat-stable/jenkins.repo
Resolving pkg.jenkins.io (pkg.jenkins.io)... 146.75.42.133, 2a04:4e42:5::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|146.75.42.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 85
Saving to: '/etc/yum.repos.d/jenkins.repo'

/etc/yum.repos.d/je 100%[===================>]      85  --.-KB/s    in 0s

2023-11-08 10:56:29 (2.87 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [85/85]

[ec2-user@ip-172-31-28-113 ~]$
```

Import a key file from Jenkins-CI to enable installation from the package:

```
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
```

```
[ec2-user@ip-172-31-28-113 ~]$ sudo rpm --import https://pkg.jenkins.io/redhat-s
table/jenkins.io-2023.key
[ec2-user@ip-172-31-28-113 ~]$
```

Sudo yum upgrade

```
[ec2-user@ip-172-31-28-113 ~]$ sudo yum upgrade
Jenkins-stable                                  160 kB/s |  27 kB     00:00
Dependencies resolved.
Nothing to do.
Complete!
```

Install Java (Amazon Linux 2023):

[ec2-user ~]$ sudo dnf install java-17-amazon-corretto -y

```
[ec2-user@ip-172-31-28-113 ~]$ sudo dnf install java-17-amazon-corretto -y
Last metadata expiration check: 0:00:44 ago on Wed Nov  8 10:58:07 2023.
Dependencies resolved.
================================================================================
 Package                     Arch   Version                  Repository   Size
================================================================================
Installing:
 java-17-amazon-corretto     x86_64 1:17.0.9+8-1.amzn2023.1  amazonlinux 188 k
Installing dependencies:
 alsa-lib                    x86_64 1.2.7.2-1.amzn2023.0.2   amazonlinux 504 k
 cairo                       x86_64 1.17.6-2.amzn2023.0.1    amazonlinux 684 k
 dejavu-sans-fonts           noarch 2.37-16.amzn2023.0.2     amazonlinux 1.3 M
 dejavu-sans-mono-fonts      noarch 2.37-16.amzn2023.0.2     amazonlinux 467 k
```

```
  libXrender-0.9.10-14.amzn2023.0.2.x86_64
  libXt-1.2.0-4.amzn2023.0.2.x86_64
  libXtst-1.2.3-14.amzn2023.0.2.x86_64
  libbrotli-1.0.9-4.amzn2023.0.2.x86_64
  libjpeg-turbo-2.1.4-2.amzn2023.0.5.x86_64
  libpng-2:1.6.37-10.amzn2023.0.6.x86_64
  libxcb-1.13.1-7.amzn2023.0.2.x86_64
  pixman-0.40.0-3.amzn2023.0.3.x86_64
  xml-common-0.6.3-56.amzn2023.0.2.noarch

Complete!
[ec2-user@ip-172-31-28-113 ~]$
```

1. Install Jenkins:

   [ec2-user ~]$ sudo yum install jenkins -y

```
[ec2-user@ip-172-31-28-113 ~]$ sudo yum install jenkins -y
Last metadata expiration check: 0:01:42 ago on Wed Nov  8 10:58:07 2023.
Dependencies resolved.
================================================================================
 Package          Architecture    Version                Repository        Size
================================================================================
Installing:
 jenkins          noarch          2.414.3-1.1            jenkins           85 M

Transaction Summary
================================================================================
Install  1 Package

Total download size: 85 M
Installed size: 85 M
Downloading Packages:
jenkins-2.414.3-1.1.noarch.rpm                  22 MB/s |  85 MB     00:03
--------------------------------------------------------------------------------
```

```
Total                                           22 MB/s |  85 MB     00:03
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Running scriptlet: jenkins-2.414.3-1.1.noarch                             1/1
  Installing       : jenkins-2.414.3-1.1.noarch                             1/1
  Running scriptlet: jenkins-2.414.3-1.1.noarch                             1/1
  Verifying        : jenkins-2.414.3-1.1.noarch                             1/1

Installed:
  jenkins-2.414.3-1.1.noarch

Complete!
[ec2-user@ip-172-31-28-113 ~]$
```

1. Enable the Jenkins service to start at boot:
   [ec2-user ~]$ sudo systemctl enable jenkins

1. Start Jenkins as a service:

```
[ec2-user@ip-172-31-28-113 ~]$ sudo systemctl enable jenkins
Synchronizing state of jenkins.service with SysV service script with /usr/lib/s
stemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable jenkins
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /
sr/lib/systemd/system/jenkins.service.
```

   [ec2-user ~]$ sudo systemctl start jenkins

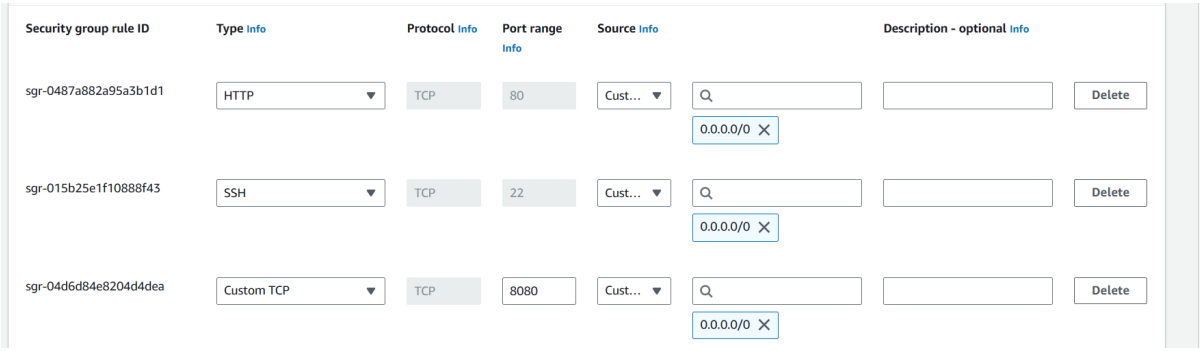
```
[ec2-user@ip-172-31-28-113 ~]$ sudo systemctl start jenkins
[ec2-user@ip-172-31-28-113 ~]$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
     Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: >
     Active: active (running) since Wed 2023-11-08 11:01:59 UTC; 15s ago
   Main PID: 28150 (java)
      Tasks: 45 (limit: 1114)
     Memory: 348.0M
        CPU: 37.968s
     CGroup: /system.slice/jenkins.service
             └─28150 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/jav>

Nov 08 11:01:30 ip-172-31-28-113.us-west-2.compute.internal jenkins[28150]: 020>
Nov 08 11:01:30 ip-172-31-28-113.us-west-2.compute.internal jenkins[28150]: Thi>
Nov 08 11:01:30 ip-172-31-28-113.us-west-2.compute.internal jenkins[28150]: ***>
Nov 08 11:01:30 ip-172-31-28-113.us-west-2.compute.internal jenkins[28150]: ***>
```

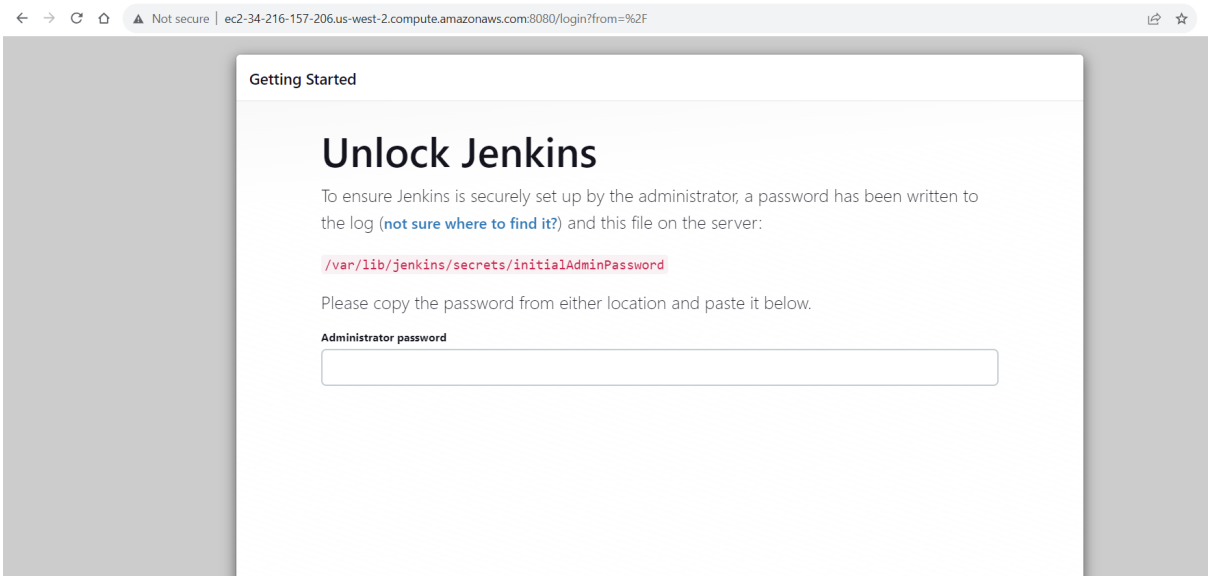
Configuring Jenkins

Jenkins is now installed and running on your EC2 instance. To configure Jenkins:

You need to open the port 8080 to enable traffic form jenkins

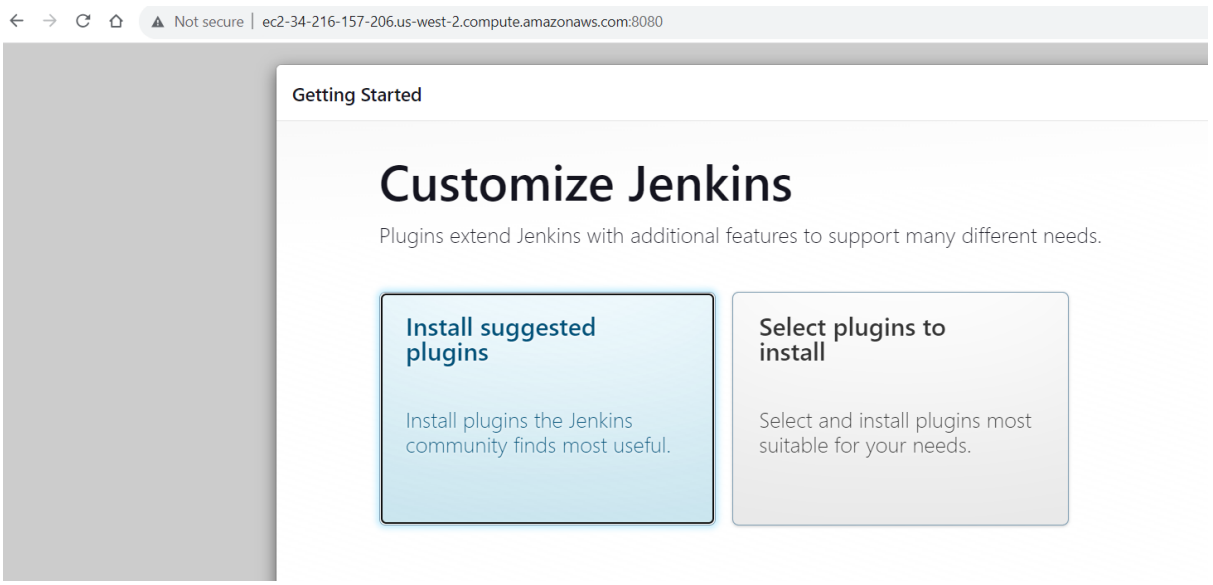


1. Connect to http://<your_server_public_DNS>:8080 from your browser. You will be able to access Jenkins through its management interface:



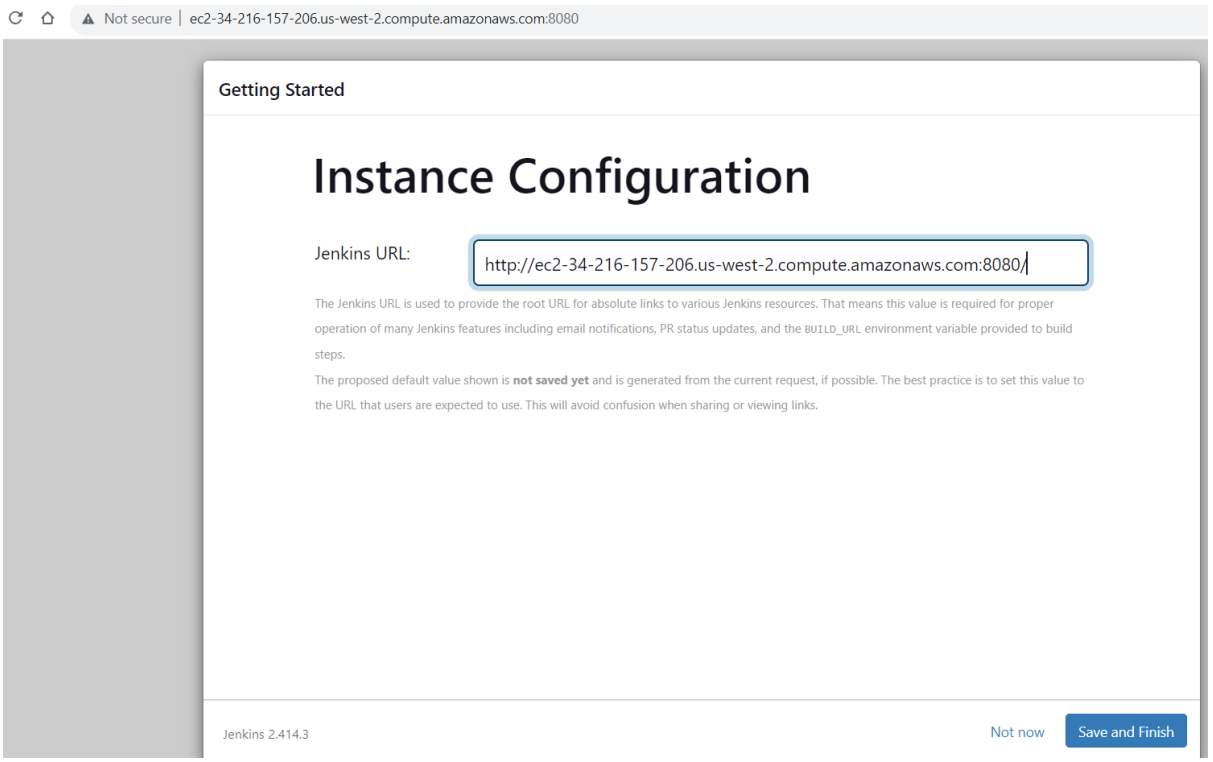
As prompted, enter the password found in /var/lib/jenkins/secrets/initialAdminPassword.

Use the following command to display this password:
sudo cat /var/lib/jenkins/secrets/initialAdminPassword



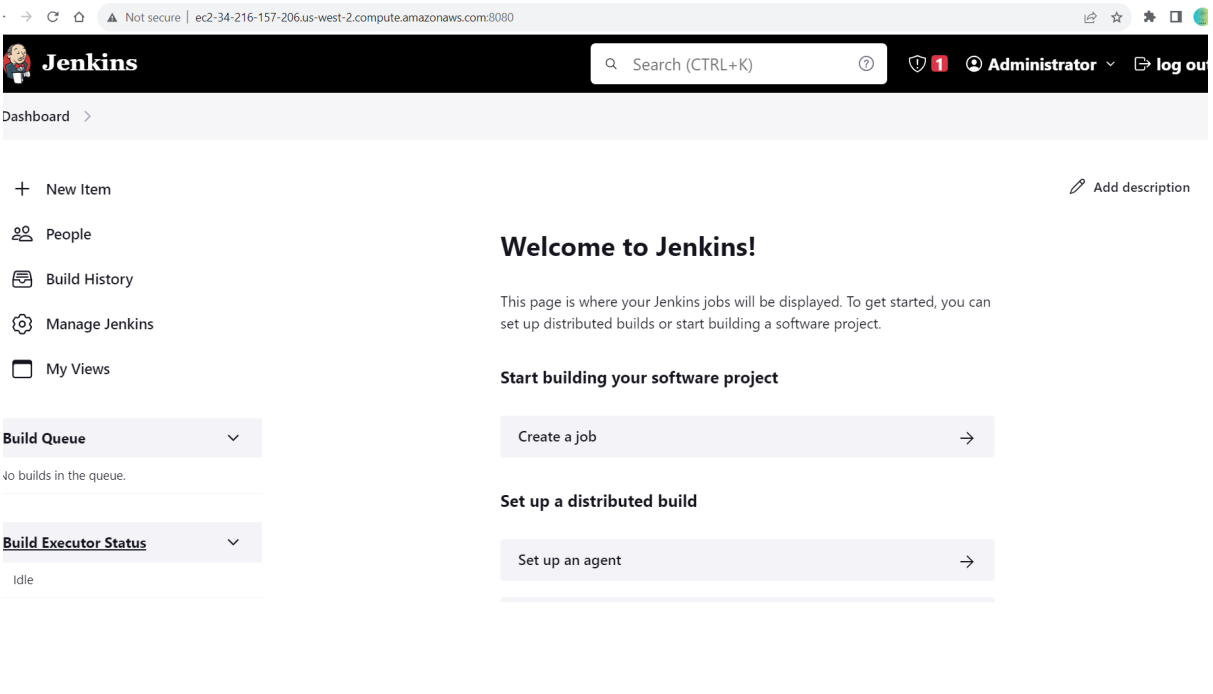
1. The Jenkins installation script directs you to the Customize Jenkins page. Click Install suggested plugins.

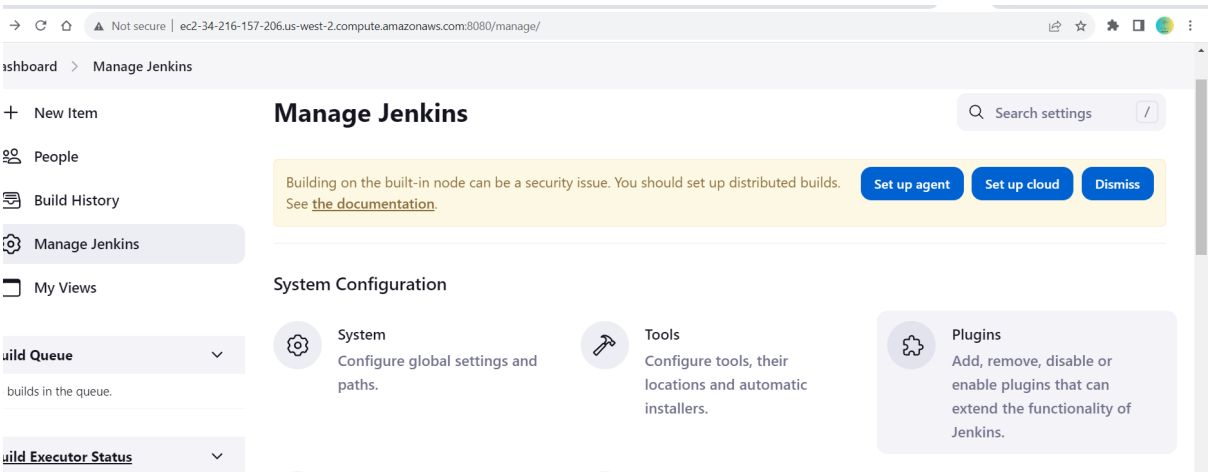
2. Once the installation is complete, the Create First Admin User will open. Enter your information, and then select Save and Continue.

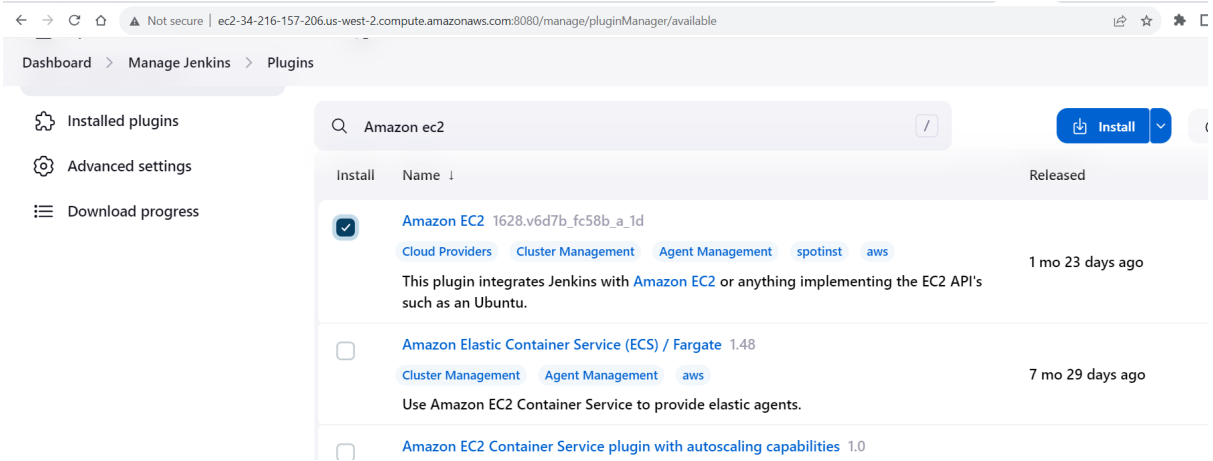


http://ec2-34-216-157-206.us-west-2.compute.amazonaws.com:8080/

1.
2. On the left-hand side, select Manage Jenkins, and then select Manage Plugins.
3. Select the Available tab, and then enter Amazon EC2 plugin at the top right.
4. Select the checkbox next to Amazon EC2 plugin, and then select Install without restart.







1. Once the installation is done, select Back to Dashboard.

2. Select Configure a cloud if there are no existing nodes or clouds.

```
[ec2-user@ip-172-31-28-113 ~]$ sudo service jenkins stop
Stopping jenkins (via systemctl):                          [  OK  ]
```